NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

AN EVALUATION OF TWO HOST BASED INTRUSION PREVENTION SYSTEMS

by

Keith Labbe

June 2005

Thesis Advisors:	Neil Rowe J.D. Fulp

REPORT DOCUMENTATION PAGE		*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.		
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2005	**3. REPORT TYPE AND DATES COVERED** Master's Thesis
4. TITLE AND SUBTITLE: Evaluation of Two Host-Based Intrusion Prevention Systems		**5. FUNDING NUMBERS**
6. AUTHOR(S) Keith Labbe		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.		
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.		**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

Host-based intrusion-prevention systems are recently popular technologies which protect computer systems from malicious attacks. Instead of merely detecting exploits, the systems attempt to prevent the exploits from succeeding on the host they protect. This research explores the threats that have led to the development of these systems and the techniques many use to counter those problems. We then evaluate two current intrusion-prevention products (McAfee Entercept and the Cisco Security Agent) as to their success in preventing exploits. Our tests used live viruses, worms, Trojan horses, and remote exploits which were turned loose on an isolated two-computer network. We make recommendations about deployment of the two products based on the results of our own testing.

14. SUBJECT TERMS Host-based Intrusion Prevention, McAfee Entercept, Cisco Security Agent, Penetration Testing			**15. NUMBER OF PAGES** 71
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

EVALUATION OF TWO HOST-BASED INTRUSION PREVENTION SYSTEMS

Keith G. Labbe
Ensign, United States Navy
B.S., United States Naval Academy, 2004

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
June 2005

Author:	Keith Labbe
Approved by:	Neil Rowe Thesis Advisor
	J.D. Fulp Co-Advisor
	Peter Denning Chairman, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Host-based intrusion-prevention systems are recently popular technologies which protect computer systems from malicious attacks. Instead of merely detecting exploits, the systems attempt to prevent the exploits from succeeding on the host they protect. This research explores the threats that have led to the development of these systems and the techniques many use to counter those problems. We then evaluate two current intrusion-prevention products (McAfee Entercept and the Cisco Security Agent) as to their success in preventing exploits. Our tests used live viruses, worms, Trojan horses, and remote exploits which were turned loose on an isolated two-computer network.   We make recommendations about deployment of the two products based on the results of our own testing.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

Without Charles Herring’s advice and assistance this thesis would not have been possible. I also owe a great debt of gratitude to Greg Abelar, Scott Cote, Reese Zomar, and the rest of the Network Security Group for all of their assistance and technical support they provided to this project. Also, thank you to my wife Tara for all of her love, support, and understanding throughout this process.

THIS PAGE INTENTIONALLY LEFT BLANK

I. HOST-BASED INTRUSION PREVENTION

A. OVERVIEW

In the beginning, system administrators were concerned with maintaining a functioning computer network. The security of that network from attacks, viruses, and other exploits, was considered secondary; the primary concern was keeping the network running. In recent years these two considerations have begun to merge. Many system administrators now acknowledge that to keep the network running, security must be a primary consideration. Tools such as firewalls, which allow only certain external-network traffic to reach the internal network, and intrusion-detection systems, which monitor network traffic for malicious activity, have become increasingly important.

In the past, traditional network-security arrangements have placed the responsibility for intrusion prevention on the firewall alone. The firewall would allow only certain traffic, as specified by policy, to pass through from the Internet to the internal network. Traditional intrusion-detection-systems merely analyzed those packets that were allowed through and examined them for unauthorized or malicious content. If unauthorized content was detected, these systems merely raised an alarm while allowing potential exploits to damage the target system(s). In recent years computer security has moved away from this traditional "warn only" approach. Host-based Intrusion-Prevention Systems (HIPS) are recently popular technologies which function more as active protectors than passive observers.

The primary goal of this research is to evaluate the idea behind host-based intrusion-prevention systems. Are the systems useful? Are they worth the added expense of purchase and management? The second goal is to make recommendations, based on the results of our testing, regarding the purchase of these products by the Department of Defense. This thesis presents the results of evaluations of two such products, McAfee Entercept and Cisco Security Agent.

Recent widespread attacks such as the Sasser and MSBlast worms, which may have infected up to 10 million computers (Lemos, 2004), have shown how vulnerable current networks are to certain attacks. Such attacks no longer require users to open email attachments; users put their systems at risk simply by connecting to the Internet with a vulnerable system. Such threats show that a reliance on signature-based intrusion-detection systems and network-based protection measures such as firewalls, is inadequate. More must be done to protect network infrastructure from exploitation.

The time between an exploit release and a vulnerability announcement, the "vulnerability threat window" (Beighton, 2004), is shrinking. Where vendors once had months to create and disseminate a patch to correct a vulnerability, they now often have only a few days. In January 2003, the Slammer worm "attacked a vulnerability that was discovered six months earlier (Moulton, 2004)"; Sasser struck in May 2004, just 18 days after the vulnerability announcement (Symantec Small Business, 2005). According to Symantec "during the first six months of 2004 the average time between the public disclosure of a vulnerability and the release of an associated exploit was 5.8 days (Sevounts, 2004)." This shrinking of the vulnerability threat window emphasizes the need for agents based on behavioral rules that act immediately to offer some possibility of protection against these emerging threats.

This thesis will discuss the concepts behind host-based intrusion-prevention systems, our testing procedures and results, and our analysis and recommendations based on that testing.

II. INTRUSION DETECTION AND PREVENTION

A. NETWORK VERSUS HOST-BASED PROTECTION

1. Network-based

Network-based intrusion-detection and intrusion-prevention systems involve one or more sensors that are responsible for monitoring the entire network (Beale, 2004). There are three common types of network-based systems. The first, and the most common, involves placing sensors into the network architecture to allow the system's Network Interface Card (NIC) to see all traffic on the network (promiscuous mode). Such systems passively monitor the network traffic. Another less common architecture requires the system to function much as a router ("inline" mode). All traffic entering and leaving the network passes through it and can be blocked or modified by it. The “pass-through” design requires a greater investment in resources, processing power, and time by the system administrators. A third type involves a single sensor node protecting a single computer ("host-based" mode) although it is situated on a network. The sensor sees only the packets bound for that specific computer (Proctor, 2001). Snort and Stealth Watch are two well know network-based Intrusion Detection Systems (NIDS).

2. Host-based

Host-based systems only monitor a single computer from within that computer. They are responsible for monitoring only the traffic that reaches their host (Beale, 2004). Host-based systems can be managed locally from the host computer, or remotely from a server running the management software. Both products tested as part of this thesis were host-based and use a management server to configure and communicate with individual host agents. Both products’ managers are capable of managing several thousand host agents from a single server.

B. HEADER/PROTOCOL INSPECTION VERSUS PAYLOAD INSPECTION

Whether network or host-based, an intrusion-detection or intrusion-prevention system has several subtasks in inspecting an arriving Internet Protocol packet.

1. Header/Protocol Inspection

Header/protocol Inspection is limited to checking information in a packet's headers. This includes such information as the source and destination IP address and what flags have been set. Header inspection requires little time and processing power and is often selected when an agent's impact on network throughput is of concern. Both of the products tested in this research include a host firewall that employs this method of inspection.

2. Payload Inspection

Payload inspection examines the contents of network packets. This level of examination typically involves "signature matching" wherein the agent searches for known strings of malicious code. This type of inspection is more thorough and effective in identifying a computer attack or exploit.

3. Decoded Packets

A third option is to do inspection at a higher level, after the packet is decoded. The agent does not inspect the individual packets; instead, it inspects the interpretation of those packets by the operating system. This prevents many common network-based obfuscation techniques such as packet fragmentation. Both of the products tested in this research employ this method of inspection.

C. STATEFUL VERSUS STATELESS PACKET INSPECTION

Agents can monitor traffic in two ways: statelessly or statefully. A stateless system is limited to making decisions based solely upon currently available information. A stateful system uses memory or other knowledge of previous activities that contributes to the quality of the decisions made. Stateful packet inspection generally requires the security system to remember recent events in a session. A network-based system must remember all of the sessions currently instantiated. This inspection requires more time, processing power, and memory, but is more effective in identifying computer exploits (About, 2005). Both agents evaluated for this research statefully monitor activities on the host computer.

D. DETECTION VERSUS PREVENTION

When the agent does identify an exploit it can respond in one of two ways.

1. Detection

An Intrusion Detection System (IDS) merely identifies and/or alerts on discovery of an exploit. Such systems often alert the network administrator to the exploit detection but take no action on their own. This lack of action has some computer security experts publicly stating that Intrusion Detection Systems are a dead technology (Messmer, 2003). However, this same lack of action also reduces the effect of false positives on the network.

2. Prevention

An Intrusion Prevention System (IPS) identifies and tries to stop the exploit before it can execute on the target computer in addition to notifying the administrator. Many protection systems, including both of those evaluated as part of this thesis, can be used in intrusion-detection-only mode should the administrator choose. This feature allows administrators more flexibility.

Both agent types, detection and prevention, typically offer the administrator different levels of protection. If the network is under attack, the administrator can increase the level of protection. However, increased protection comes at the price of raising the ratio of false positives (false alarms) to false negatives (successful exploits). In the case of a protection system such an increase can introduce self-inflicted denial of service problems.

E. SIGNATURE VERSUS BEHAVIOR BASED DETECTION

Current monitoring systems fall into four main categories with regard to how recognition of exploits is achieved: signatures, behavior, combinations, and anomalies.

1. Signatures

Signature-based systems use a database of known attack signatures to identify potential exploits. In order for these systems to work, exploits must be identified by some other means first to have their signatures isolated and disseminated. During the period between the release of the exploit and the dissemination of its signature, any network protected exclusively by a signature-based system is vulnerable to that exploit.

If a new variant of that exploit is released that does not match the exact signature, the process must begin again. And a non-malicious program that matches the signature is falsely identified by the system as an exploit.

2. Behavior

Agents based on behavioral rules use a database of rules that describe normal behavior. Any program that violates these rules of behavior is identified by the agent as an exploit. Because the systems rely on a pattern of behavior (such as a sequence of commands) rather then an exact signature, they do not require an exploit to first be identified and a signature disseminated. Such systems offer protection against some exploits that have not been invented yet, referred to as “zero-day” protection. Systems based on behavioral rules do not require the frequent updates that signature-based systems do; however, they still require some updates as exploits change their behavior with time in order to avoid detection. The Cisco Security Agent relies on such behavioral rules. Systems based on behavior rules cannot provide an exact identification of an exploit used against them, just a category.

3. Combinations

Combination systems use both attack signatures and behavioral rules. Such systems offer the “zero-day” protection of a behavior-based system while also providing the attack recognition of a signature-based system. McAfee Entercept is a combination system.

4. Anomalies

Anomaly-based systems are typically used as network intrusion-detection Systems. The systems compare current network traffic with typical (baseline) network traffic and locate any statistical anomalies (deviations from baseline) (Stallings, 2003). Sufficiently anomalous traffic is identified as a possible exploit.

F. SUBVERTING SIGNATURE-BASED SYSTEMS

Two options exist for circumventing signature-based monitoring systems. The first is to use an exploit that does not have a signature. This can be done by creating a new exploit or by modifying an old exploit so that the signature no longer matches. This option requires some expertise and time, and only a few attackers can provide this.

The second option against network-based agents is to modify the protocol in such a way that the system cannot see the entire signature of a known exploit as one observable string. A recent paper (Rubin, 2004) discussed a program called AGENT (Automatic Generation for Network Intrusion Detection System Testing tool) that "mutated" the sequences of several known attacks and tested against Snort, a well-known signature-based network intrusion-detection system. The mutations included fragmentation, retransmission, and header changes. For each attack attempted the researchers could generate at least one mutated session that Snort did not detect. But this approach is less likely to succeed against a host-based system.

G. SUBVERTING NETWORK-BASED SYSTEMS

Ptacek and Newsham (Ptacek, 1998) argue that current network-based systems are inherently flawed in three ways.

1. Insertion

One kind of attack relies on the monitoring system accepting a packet that the victim computer rejects. Because of differences in how different computers handle different Internet packets, it is not difficult to create such a packet. Such packets could include malformed or incorrect header information and improper fragmentation. When the intrusion-detection system examines the session it does not detect the exploit because of the camouflage; however, the victim computer rejects the inserted packet and is successfully exploited.

Unless the network segment being monitored by the system is completely homogenous or incredibly small, it is not possible to configure it to reject every packet the victim computer rejects because that would require ultimately executing the code itself. Determining whether code is safe is an undecidable problem similar to the undecidable "Halting Problem" for automata.

2. Evasion

Evasion relies on the monitoring system rejecting a packet that the victim computer ultimately accepts (the exact opposite of insertion). When the intrusion-detection system examines the session no known attack signatures are found, while the victim computer is successfully exploited. This type of attack is only effective against systems that rely on passive analysis of network traffic. Since the monitoring system is merely eavesdropping on the network it cannot prevent the victim computer from accepting the same packet that it rejected; while a system that monitors “in-line” would be able to discard the rejected packet, thereby preventing it from reaching the host system

3. Denial of Service

A third means of subversion involves a denial-of-service attack against the monitoring system itself. This attack is particularly effective against systems that rely on passive analysis, but could also function against others. If the attacker can crash the monitoring system, it will not be able to monitor network traffic. If the system “fails open”, the unmonitored traffic continues to flow to and from the network leaving it vulnerable to exploit. A system designed to “fail closed” would cause a self-inflicted denial of service, though it would halt any attempt to exploit the network.

4. Insider Threats

A fourth vulnerability of network-based monitoring systems is their potential to miss exploits conducted by someone inside the network. If the system monitors only the external network, it cannot detect exploits launched from one computer to another across internal network connections.

III. ADVANTAGES OF HIPS PROTECTION

A. ADVANTAGES AT THE NETWORK LEVEL

1. Defense in Depth

Two lines of defense are always preferable to one in warfare, cyber or otherwise. Host-based intrusion-prevention systems add an additional layer of protection beyond network-based ones. This added layer of protection increases the chances of stopping an attacker before he can successfully exploit any computers on a network. The remaining unexploited computers are still protected and must be individually exploited increasing the chances his attacks will be detected.

2. Reduced Perimeter Dependence

Traditional network security architectures call for a perimeter, or boundary, to separate segments of the network from the Internet and from one another. As you move further into the proprietary network and further away from the public Internet, the access controls between network segments generally increases (become increasingly selective/restrictive). This traditional architecture works well on any network that can be divided into segments with well-defined perimeters. However, as the U.S. Department of Defense moves increasingly towards "network-centric" operations and the architecture known as the Global Information Grid, this type of network architecture will no longer be possible. The Global Information Grid “will provide authorized users with a seamless, secure, and interconnected information environment (National Security Agency, 2005)” as a world-wide, dynamic, ad-hoc network. The network will contain everything from hand-held computing devices carried by Marines in the field, to ship-based servers supporting an entire carrier battle-group. The burden of protection will move increasingly to the edge of the network and be borne by each individual host. Host-based Intrusion-Prevention Systems offer administrators an effective means of protecting those hosts.

3. Distributed Sensors

Each host-based agent of a host-based intrusion-prevention system acts as a sensor on the network. This feature provides the network much greater capability for detecting and preventing attacks. Instead of receiving data from a few strategically located sensors on the network, the administrator receives it from every host. Even if an attacker is able to circumvent parts of the network, they will not be able to evade the host-based protection located on each individual system.

B. ADVANTAGES AT THE HOST LEVEL

1. Protection against Local Threats

Host-based intrusion-prevention systems protect the host from any malicious activity, not just network based malicious activity. A network-based protection system cannot prevent a user from executing local attacks such as privilege escalation, whereas a host-based system can.

2. Centralized Management

Usually a host-based intrusion-prevention system's management server allows an administrator to configure and manage all host-based agents remotely. The management server allows the administrator to view all alerts for the network. Malicious-activity reports can include detailed forensics information about the alert, allowing the administrator to evaluate if the host was actually attacked.

3. Tailored Protection

Host-based agents allow the administrator to tailor protection to each individual host. These configurations can be handled in groups, such as all email servers or all remote users, or by individual hosts. Such an architecture can provide greater protection for mobile hosts, and more consistent protection for ad-hoc wireless networks, networks wherein the perimeter is difficult to establish and changing often.

C. ADVERTISED HIPS FUNCTIONALITY

Two commercially available host-based intrusion-prevention systems were evaluated as part of this research. Each product advertises a number of functional features.

1. McAfee Entercept

Information described in this section was obtained from the Entercept 5.0 Evaluation Guide. We do not specifically endorse or deny any claim found in this section. Our own results can be found in chapter 5 of this thesis.

MacAfee Entercept claims to use a combination of behavioral rules, attack signatures, and a process firewall to protect against “known and unknown malicious activity including, but not limited, to worms, Trojan horses, buffer overflow attacks, malformed commands, critical system file modifications and privilege escalation (McAfee, 2005)”. To provide this protection, Entercept inserts itself into the system-call chain, using a kernel level driver, and redirects the entries in the system-call table to the Entercept driver. It is then able to intercept “select system calls and API calls before the OS executes them (McAfee, 2005)”. When an application requests a file, Entercept checks the request against its behavioral rules and signatures and only allows those requests that it deems to be non-malicious.

The process firewall, available only for Windows platforms, gives administrators the ability to control traffic to and from individual systems. They can create firewall rules that control access to network resources. These rules can be configured as incoming, outgoing, or both. The firewall also includes a network engine and a set of signatures that inspect for exploits in the communications layer. If a malicious packet is detected, the “network engine discards it before it is processed by the TCP/IP [Transmission Control Protocol/Internet Protocol] stack (McAfee, 2005).” Any alerts generated by the firewall are reported to the management server.

Entercept’s hybrid approach allows it to claim to prevent both known and zero-day (previously unknown) attacks. Instead of merely preventing an exploit from propagating, Entercept’s hybrid approach allows it to prevent it from compromising the service. Then the host system does not need to be rebooted, thus preventing any denial of service. The hybrid approach also lowers the number of false alarms generated by the host agents, with the behavioral rules minimizing false positives while the signature database minimizes false negatives. Entercept’s protection against zero-day attacks “reduces the need for immediate patch deployment—enterprises can deploy patches after careful research and testing… (McAfee, 2005)”.

Entercept’s comprehensive protection also includes buffer overflow protection. It claims patented technology that protects the host by preventing code executions resulting from buffer overflows. Entercept protects system resources by “locking down critical system resources (specific system files, settings, registries keys, services, etc.) (McAfee, 2005)". Protection against privilege escalation prevents attacks from gaining root-level privileges. Entercept can also provide detailed forensic information about an exploit.

The management server and the management console allow the administrator to quickly and easily view all alerts and attacks, configure the host agents, and perform data analysis. The management server can control up to 10,000 agents. The management server also allows the users of the host agents to create their own firewall and behavioral rules as necessary.

Entercept agents are available in four variants: standard edition, Web-server edition, database-server edition, and a Web/database combination. Standard editions are available for Windows, Solaris, and HP-UX operating systems. Web-server editions are available for IIS 4, IIS 5, IIS 6, Apache 1.3.6 and higher, Apache 2.0.42 and higher, iPlanet 4.0 and 4.1, and Sun ONE 6.0. Database-server editions are available for Microsoft SQL server 2000 only.

2. Cisco Security Agent

Information in this section was obtained from the Cisco Security Agent version 4.5 Data Sheet. We do not specifically endorse or deny any claim found in this section. Our own results can be found in chapter 5 of this thesis.

The Cisco Security Agent uses behavioral rules to proactively defend its host computer from damage through all five phases of an attack. It is designed specifically to thwart attacks with no known signature. The Cisco Security Agent also provides "host intrusion prevention, a distributed firewall, malicious mobile code protection, operating system integrity assurance, and audit log consolidation; all within a single agent (Cisco 2005)".

Cisco identifies five key phases to a computer attack. The first is the Probe or reconnaissance phase, during which the attack gathers as much data as possible about the target computers and their network's topology. Next, during the Penetration phase, the attacker utilizes exploits such as buffer overflows and email attachments to gain access to a targeted system. During the third, or Persistent phase, the attacker installs back doors and updates system registries to allow her access to the computer at a later date. During the fourth, or Propagate, phase the attacker extends control to other computer systems on the network. The fifth and final phase, Paralyze, involves the attacker disrupting services on the targeted network.

The Agent resides between the application and kernel level of the host system. Cisco claims this architecture minimizes the impact on the stability of the operating system and allows the agent to intercept all system calls to file, network, and registry sources, as well as to dynamic run-time resources such as memory pages and shared library modules (Cisco, 2005). The behavior of the intercepted calls is correlated with other calls and evaluated against a set of rules that define acceptable behavior. Decisions about whether to allow or deny these system calls are then made in real time. Since protection is based on blocking malicious behavior, the agent blocks both known and unknown attacks without requiring updates. Correlation of system-call behavior is performed on both the host agent and the management server, thus allowing the system to effectively identify and block malicious host-based activity and global attacks such as network worms or distributed scans.

The management server is accessed through a Web-browser interface and requires "Cisco Works" in addition to the management-server software. A single management server can control up to 20,000 agents. However, using the provided installation guide, three servers can control up to 100,000 agents. All alerts generated by the agents are reported to the management server. If an agent is disconnected from the management server at the time an alert is generated it stores the alert and reports it to the server when connection is reestablished. If contact with the management server is lost for any reason, all agents will continue to function at the last defined protection level, allowing remote users to receive the same level of protection.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. TEST SETUP AND METHODOLOGY

A. METRICS OF EFFECTIVENESS

1. False Negatives

One useful metric is the number of false negatives (missed attacks). In order to measure these metrics we used two remote penetration tools and a number of malicious code exploits. The actual exploits used in this research are listed in section C of this chapter.

a. Core Impact

A successful Core Impact exploit installs a level zero agent on the victim computer. A "level zero agent" is specific to Core Impact. It returns administrator level privileges to the attacking computer. Once a level zero agent has been uninstalled it leaves no trace on the victim computer. The exploits were allowed to run until Core Impact reported success or failure. The management server was then checked for any alerts.

b. Metasploit

A successful Metasploit exploit performs a number of payload actions as specified by the user. Two payloads were used for this thesis: "Add User" which adds an administrative user to the victim computer, and "Win Bind" which returns a command shell from the victim computer to the attacker. The exploits were allowed to run until they reported success or failure. The management server was then checked for any alerts.

c. Malicious Code Installation

Three types of malicious code were used in this research: viruses, worms, and Trojan horses. A successful malicious-code exploit either spawns a process on the victim machine and that process continues to run until manually stopped, or the process causes some noticeable action on the victim machine that could only be attributed to the exploit. Noticeable actions include, but are not limited to, mass mailing, self-deletion, and the creation of new files on the victim machine. These exploits were allowed to run for approximately two minutes, unless the process started by the exploit was still running. IF the process continued to run it was given another two minutes. Processes were monitored through the windows “Task Manager”. After the allotted time, the management server was checked for any alerts and any still-running processes were halted.

2. False Positives

Another useful metric is the number of false positives (normal activities erroneously identified as malicious). An event was deemed to be a false positive if an alert was generated by the agent that was not caused by an exploit or malicious code test. There were no specific tests aimed at eliciting false positives, however we kept a record of any that occurred in the course of our tests for overall product evaluation purposes.

3. Impact on Protected System Throughput

While there were no metrics designed to measure the effect on system throughput, we were able to make some general observations based on the amount of time required by the agent to perform a given task. Tasks included things like loading the management server and opening programs such as the Web browser used during the testing of the software.

B. TEST LAB SETUP, CONFIGURATION AND RATIONALE

1. Topology and Component Selection

Two computers were used to perform the tests. The computers were networked together using a crossover cable; no other network connections were used. This configuration allowed us to use live exploits without fear of accidental infection of other computers on the network. This configuration also isolated our computers from non-test network activities and traffic. One computer, *Inside*, was used as the victim machine throughout the testing. The other computer, *Outside*, served as the management server for the HIPS agent, the attacking computer, and the email server.

2. Computer Configuration

The victim computer--*Inside*--used the Microsoft Windows 2000 Advance Server Service Pack 0 operating system. We also installed SQL 2000 and IIS 5.0. These service packs and versions were purposely chosen because they are out of date. This ensured a large number of possible vulnerabilities. We used Norton Ghost to create an image of the victim machine. This image was used to restore the victim machine as necessary to its original baseline configuration. After restoration, the agent currently being tested was reloaded onto the machine as required to continue the testing with remaining exploits.

The attacking computer--*Outside*--also used the Microsoft Windows 2000 Advanced Server operating system. Service packs were only installed as required by the HIPS management server. McAfee Entercept required service pack 2. Cisco Security Agent required service pack 4. Core Impact and Metasploit were also installed onto *Outside* in order to conduct remote exploits. We also installed an email server on *Outside*. No aspects of the email server itself were tested; it was merely used to send exploits from our attacker to the victim machine.

3. Configuration Profile of both Test HIPS

McAfee Entercept version 4.056 was tested at Level 2 Protection as defined by its management console. All high and medium alerts were blocked by the agent and logged by the management server. All exploits identified as false negatives were then tested a second time at Level 3 Protection as defined by the management console. Level 3 Protection blocks all high, medium, and low alerts and logs all high, medium, low and information alerts with the management server. During the reconnaissance phase the firewall was tested both in warning mode and protection mode.

Cisco Security Agent version 4.5 was tested at the medium security level as defined by the host agent. All exploits identified as false negatives were then tested a second time at the high security level as defined by the host agent. The predefined rule modules “all servers” and “IIS servers” were enforced on *Inside*.

4. Identification and Isolation of Control Variables

To check that all of the exploits worked properly, we first tested them against an unprotected image of the victim machine. Since all the exploits did work under these circumstances, any exploit that failed to infect/penetrate *Inside* after the protective agent was installed was logically assumed to be attributable to the success of the Intrusion Prevention System.

5. Assumptions and Limitations of Test

Neither agent was tested against an incoming threat from a peer-to-peer program such as ICQ or Kazaa. It was decided that installing and configuring our own peer-to-peer program would be too difficult and time consuming for this research.

Effects on system throughput were not tested as part of this research. Such tests were beyond our scope due to the large number of computers required. Future work in this area is recommended.

C. TESTING PHASES

As discussed earlier, the first step in many hacker attacks is reconnaissance of the target system and the network it resides on. Thus, the first phase of our testing involved evaluating the agent's ability to thwart a potential attacker's reconnaissance efforts. Success in this area may be considered security through obscurity, but we should not disregard its potential value. Decreasing the amount of information a hacker can easily gather about a network reduces their chances of success. Early awareness of reconnaissance activity may also provide the target system's owner with valuable lead time with which to employ additional protective/deterrence measures.

The remaining phases of testing involved evaluating the agent's ability to block an exploit once it has been launched. While many exploits can be used against a computer, almost all fall into one of five main categories: remote, email, Web-page, disk, and peer-to-peer. For this evaluation we tested exploits in the remote, email, Web-page, and disk categories. For each of these four categories, twelve attacks were used, for a total of forty-eight total test cases.

The exploits were chosen to cover as broad a range as possible and included worms, Trojan horses, and viruses. We tried to select as equal a ratio as possible from each category. However, after our tests against an unprotected system a large number of our selected Trojan horses were eliminated because they did not meet our standards of success. Additionally, although neither product claimed to stop viruses, we wanted to see how they would perform against them. Exploits were also chosen with regard to age, we did not want all of our exploits to be very old or very new, we tried to choose a number of new and old exploits. If we chose only very new exploits we would not know if the products still blocked older well known exploits. If we chose only very new exploits we would not know if the products were effective against new exploits. Thus a mix of exploits was chose, some old and some new. The remote exploits chosen attacked a number of different services as well as core Windows components.

We divided the evaluations into five phases, one phase for each category, and the reconnaissance phase. Each phase of testing was completed for each exploit before moving on to the next phase. Exploits are listed below in the order in which they were tested during both evaluations. The testing procedure is given in Appendix A.

1. Reconnaissance

The reconnaissance phase was conducted using Super Scan 4. The scans were conducted from *Outside* to *Inside*. McAfee Entercept was tested twice, once with the firewall in protection mode, and once with the firewall in warning mode. Warning mode still generated alerts but did not block malicious behaviors. Cisco security agent was tested once. The tests was conducted only once against Cisco Security Agent because we could not find a way to disable the firewall capability without disabling the agent itself. The tests were also carried out against an unprotected system for comparison purposes.

2. Remote Exploits

The remote-exploit phase was conducted using Core Impact and Metasploit. The exploits were launched from *Outside* against *Inside*. The following exploits Core-Impact exploits were evaluated:

IIS CGI Filename Decode

IIS Unicode

IIS IDS-IDQ

SQL Server Hello

MSRPC DCOM

SQL Server CAN-2002-0649

IIS ASN.1 Big String SpNeGo

MSRPC LSASS Buffer Overlow

The following Metasploit exploits were evaluated:

iis_nsiislog_post – win bind payload

iis50_printer_overflow – win bind payload

iis50_webdav_ntdll – win bind payload

msrpc_dcom_ms03_026 – add user payload

3. Email Exploits

For the email phase, an email with an attached exploit was generated on *Outside* and sent to *Inside* via the email server. Microsoft Outlook Express was then used to view the message on *Inside* and attachments were opened. The malicious code was then executed on *Inside*. The following exploits were used:

Iworm.lovegate.i

Iworm.Loveletter

Iworm.Klez.h

Iworm.Moodown

Iworm.Navidad.b

Iworm.Netsky.d

Worm.Win32.Chainsaw.a

Worm.win32.Donk.c

Backdoor.SdBot.aa

Win2k.inta.1688

Iworm.Radix

Iworm.Mydoom.g

4. Web Page Exploits

For the Web-page phase, a Web page was created on *Outside* that hosted the exploit. Using the Web browser Internet Explorer on *Inside*, we browsed to the page and executed the exploit. Internet Explorer provides two options for downloading and executing Web-based content, "run this program from its current location", and "save this program to disk", and we tested both. The following exploits were evaluated:

Netbus Trojan version 1.7

Trojan.Win32.virtualroot

IIS-Worm.CodeGreen.a

Willow.2013

Win2k.Stream

Win32.Cabanas.b

Win32.Ghost.1667

Win32.HLLO.Zori

Win32.Lash.d

Win32.Matrix.Ordy.a

Win32.Redemption.b

Iworm.Mydoom.h

5. Disk Exploits

For the disk phase, the exploit was placed on a 3.5” floppy disk. The disk was then inserted into *Inside* and executed.

IIS-Worm.IIS Worm

Worm.Win32.Lovesan.a

Worm.Win32.Muma.C

Worm1

Worm2

Win32.Small.2280

Iworm.Aliz

Worm.win32.sasser.b

Worm.win32.welchia.g

I-worm.bagle.at

Trojan.call911

Iworm.alanis

D. EXPLOIT CODE EMPLOYED

1. Malicious versus Non-Malicious Code

Because both systems rely at least partially on behavioral rules as a means of detecting exploits, only complete examples of malicious code were used in this research. Simulated malicious code may contain the complete signature, but by its very nature it does not mimic all of the characteristic behaviors of malicious code. However, complete malicious code is dangerous and must be used only in an isolated environment such as ours.

2. Malware Code Source

The Netbus Trojan was downloaded from Hackers Playground (Hackers Playground, 2001). All other virus, Trojan, and worm exploits were obtained from or created with tools available on VX Heavens (VX Heavens, 1999). Worm One and Worm Two were created specifically for this research using the P0ke's Worm Generator. Malicious code names listed in this thesis are as they appear on VX Heavens.

3. Remote Exploit Code Source

Core Impact is a commercially available automated penetration testing tool. Core Impact version 4.0.1 was used for this research. It can be purchased from Core Security, <http://www.coresecurity.com>.

Metasploit is an open-source penetration-testing tool available over the Internet. The Metasploit framework version 2.3 was used for this research. Metasploit can be downloaded from the Metasploit project homepage, <http://www.metasploit.com>.

4. Reconnaissance Testing Code Source

Super Scan is a Transmission Control Protocol port scanner. Super Scan version 4 was used for this research. Super Scan can be downloaded free from Foundstone <http://www.foundstone.com>.

E. PROBLEMS ENCOUNTERED DURING TESTING

1. McAfee Entercept

Two special problems were encountered while testing McAfee Entercept. The first occurred after I locked both computers rather then shutting them down for the night. On the next day the management server was no longer able to communicate with the agents. I then attempted to reinstall the management server but was unable to because Entercept's agent is self-protecting. The agent on the management-server computer would not allow the management server to be uninstalled since it was in protection mode. Because the management server could no longer connect to the agent I could not take it out of protection mode to perform the reinstall. After several attempts I was able to successfully uninstall both the agent and the management server by restarting the computer in safe mode. However, this did not allow the management server to properly uninstall the database. As a result, when I attempted to reinstall the management server I encountered numerous server-agent consistency errors. After manually deleting the database, registry values, and other files associated with the database I was able to successfully reinstall the management server. To ensure this problem did not happen again I properly shut down both computers at the conclusion of each day and did not put the agent on the management server in protection mode. Leaving the management server unprotected did not affect our test results, but it is not recommended for real world operations.

The second problem occurred after Entercept testing was completed. While attempting to uninstall the management server and database I mistakenly uninstalled the database first. When the management server tried to uninstall, several errors occurred as it tried to uninstall the database. After several attempts I decided to simply delete the management server manually rather then asking it to uninstall itself. To avoid this problem, I recommend uninstalling the management server first followed by all traces of the database that remain.

2. Cisco Security Agent

Two special problems were encountered while testing Cisco Security Agent. The first occurred during our initial installation. At some point during our testing several of the Windows Operating system files on *Outside* were corrupted. This corruption did not affect *Outside* in any way, and went completely unnoticed until we attempted to install the Cisco Security Agent management server. When the management server is installed a Microsoft SQL 2000 database is installed on the computer hosting the management server. Because of the corrupted files this database was not installed correctly. However, we did not know about the corrupted files and thought the incorrectly installed database was the problem. Eventually with the help of some tech support from Cisco we discovered that the bad database was a symptom of the problem and not the actual problem. Once a new version of windows was installed the installation of Cisco Security Agent proceeded without any further difficulties.

The second problem also occurred during installation. During the course of our testing it was necessary on several occasions to return *Inside* to its baseline configuration. This was accomplished using a previously compiled image of *Inside*. After the image was restored we then reinstalled the Cisco Security Agent host agent on *Inside*. After the installation the computer rebooted and attempted to register with the management server. Unfortunately, the agent was unable to do so. After numerous restarts, and other attempts to register the host agent with the management server it would eventually register. We was not satisfied with how long it took the agent to register and attempted to discover why it would not register immediately after restart. The agent log file listed a specific error number, unfortunately for us we did not know what that number meant. After about half an hour of testing we discovered the problem was caused by the management server already containing a host agent named *Inside*. If this host was deleted from the management server the registration proceeded without a problem. If the host record was not deleted it took the management server several minutes to recognize that this was in fact the same host and it should be allowed to register.

V. TEST RESULTS

A. RECONNAISSANCE PHASE

The Reconnaissance Phase testing was conducted using SuperScan 4.0. SuperScan can be used by both administrators and hackers. It provides the user with information about the individual hosts on the network and is a useful tool for those attempting to attack or defend a computer network. The first test was a normal port scan using SuperScan's default settings. The second was a Windows Enumeration Scan using SuperScan's default settings.

Seventeen different tests provided by SuperScan were evaluated. The first three tests (TCP Ports, UDP Ports, and Banner Grabbing) were done using a port scan. Port scans can be used against any computer regardless of the operating system. The remaining tests were done from the Windows Enumeration scan and are only effective against the Microsoft Windows operating System. "TCP Ports" indicates the number of Transmission Control Protocol ports that SuperScan found open on the victim computer, and "UDP Ports" indicates the number of User Datagram ports. "Banner Grabbing" indicates additional data about both types of open ports, including information about the services running such as "Http 1.1" or "Microsoft IIS 5.0". "Name Table" indicates that SuperScan could discover the names, such as Inside, associated with the victim computer. "MAC Address" indicates SuperScan could discover the victim computer's unique Media Access Control address. "Workstation Type" indicates that SuperScan could determine what operating system the victim computer was running. "Users" indicates that SuperScan could provide the names of all the victim computer's users along with data such as the last time the user logged on, when the user last changed his password, and when the current password expires. "Groups" indicates that SuperScan could provide the names of all the groups on the victim computer and users associated with each group. "RPC Endpoints" indicates that SuperScan could determine information about the Remote Procedure Call pointers. "Password Policy" indicates that SuperScan could determine the password policy of the victim computer such as how many incorrect logon attempts are allowed and the minimum password length. "Shares" indicates that SuperScan could find at least one file or drive being "shared" by the Windows operating system of the victim computer. “Time of Day” indicates that SuperScan could determine the system time on the victim computer. “Logon Sessions” indicates that SuperScan could determine who was logged on to the victim computer, how long they have been logged on, and how long they have been idle. “Drives” indicates that SuperScan could discover the drive letters, such as “C”, currently being used on the victim computer. “Trusted Domains” indicates that SuperScan could determine what domains the victim computer belonged to. “Services” indicates that SuperScan could determine what services were currently installed on the victim computer and whether or not the service was currently running.

Table 1 summarizes the results of testing, where "yes" means that SuperScan succeeded in its reconnaissance (and thus the protection mechanism failed). Analysis of McAfee Entercept’s performance during this phase is in section B.9 of this chapter, and analysis of Cisco Security Agent’s performance in section C.9. SuperScan’s complete reports are attached as Appendix B.

	Unprotected	Entercept Firewall Off	Entercept Firewall On	Cisco
TCP Ports	Yes – 10	Yes - 10	Yes – 5	Yes - 10
UDP Ports	Yes – 3	Yes – 3	Yes – 1	Yes - 3
Banner Grabbing	Yes	Yes	Yes	Yes
Name Table	Yes	Yes	Yes	Yes
MAC Address	Yes	Yes	Yes	Yes
Workstation Type	Yes	Yes	Yes	No
Users	Yes	Yes	Yes	No
Groups	Yes	Yes	Yes	No
RPC Endpoints	Yes	Yes	Yes	Yes

	Unprotected	Entercept Firewall Off	Entercept Firewall On	Cisco
Password Policy	Yes	Yes	Yes	No
Shares	Yes	Yes	Yes	No
Time Of Day	Yes	Yes	Yes	No
Logon Sessions	Yes	Yes	Yes	No
Drives	Yes	Yes	Yes	No
Trusted Domains	Yes	Yes	Yes	No
Services	Yes	Yes	Yes	No

Table 1. Reconnaissance Phase Results

B. TESTING EXPLOITS AGAINST MCAFEE ENTERCEPT

1. Remote Phase

McAfee Entercept prevented 11 of our 12 remote exploit attempts. The only failure was for Metasploit's MSrpc_dcom_ms03_026 exploit, using the "add user" payload. Using this payload we were able to add an administrator level account onto the victim machine. We are unsure as to why this occurred: Entercept blocked "Core Impacts" use of the same exploit. We ran the test again with the same exploit and a different payload (Win32_bind); this time Entercept blocked the exploit. We then ran the test a third time, this time again with the "add user" payload, and exploited the victim computer. The exploit was then run again at the highest protection level. This time Entercept could block the exploit by blocking access to the command prompt.

Another exploit, "IIS IDA-IDQ", was unable to execute, but the management server did not record any alerts. We are unsure as to why this occurred. In our testing against an unprotected system the exploit succeeded.

The exploits used and the alerts generated by host agents are listed below in Table 2. A "N" in the FN, or False Negative, column indicates that the exploit was blocked by the Entercept agent; a "Y" indicates a false negative, or missed attack. All columns containing a “Y” also contain a second “Y” or “N”; indicating the result of a second test performed at the highest security level. Alerts generated during only the highest tested are marked with as “(low)”.

	Tool	Exploit	FN	Alerts
1	CI	IIS CGI Filename Decode	N	1 IIS Directory Transversal and Code Execution 1 IIS Remote Command Execution 1 IIS Directory Traversal
2	CI	IIS Unicode	N	1 IIS Directory Traversal and Code Execution
3	CI	IIS IDA-IDQ	N	0
4	CI	SQL Server Hello	N	1 Generic Buffer Overflow
5	CI	MSRPC DCOM	N	1 svchost Buffer Overflow (RPC DCOM)
6	CI	SQL Server CAN-2002-0649	N	1 Generic Buffer Overflow
7	CI	IIS ASN.1 Big String SpNeGo	N	1 Generic Buffer Overflow
8	CI	MSRPC LSASS Buffer Overflow	N	1 Generic Buffer Overflow
9	CI	iis_nsiislog_post	N	1 IIS envelope modified by IIS Process (Medium)
10	CI	iis50_printer_overflow	N	1 IIS printer extension request
11	CI	Iis50_webdav_ntdll	N	2 IIS webdav Buffer Overflow
12	CI	Msrpc_dcom_ms03_026	Y N	1 CMD Tool Accessed (Low)

Table 2. McAfee Entercept Remote Phase.

FN = False Negative

CI = Core Impact MS = Metasploit

2. Email Phase

McAfee Entercept prevented only 2 of our 12 email exploits. False negatives included Iworm.navidad.b, which began emailing itself through our email server to the email accounts listed in *Inside's* address book, and Iworm.mydoom.g, which eliminated the user's ability to view the task manager.

When the exploits were retested against the highest level of protection, Entercept still failed to block any of them. Although Entercept generated a "new startup program creation" alert for many of them during this retest, it did not prevent most of them from remaining persistent after restart, including Iworm.navidad.b which set out 73 emails in less then a minute, and Iworm.Loveletter which prevented the host agent from starting again after restart. The exploits used and the alerts generated are listed below in Table 3.

	Exploit	FN	Alerts
1	Iworm.lovegate.i	Y Y	1 New startup program creation (low)
2	Iworm.Loveletter	Y Y	1 New startup program creation (low)
3	Iworm.Klez.h	N	3 Agent Shielding File Modification (Shutdown attempt) 1 System drive Executable Modification (Medium)
4	Iworm.Moodown	Y Y	1 New startup program creation (low)
5	Iworm.Navidad.b	Y Y	1 New startup program creation (low)
6	Iworm.Netsky.d	Y Y	1 New startup program creation (low)
7	Worm.win32.Chainsaw.a	Y Y	1 New startup program creation (low)
8	Worm.win32.Donk.c	Y Y	1 New startup program creation (low)
9	Backdoor.SdBot.aa	Y Y	0

	Exploit	FN	Alerts
10	Win2k.inta.1688	N	1 System File Modification in Root Drive 1 System Drive Executable Modification
11	Iworm.Radix.	Y Y	1 New startup program creation (low)
12	Iworm.Mydoom.g	Y Y	1 New startup program creation (low)

Table 3. McAfee Entercept Email Phase.

FN = False Negative

3. Web Phase

McAfee Entercept prevented all 12 of our 12 web page exploits for our first test. In the first test we selected the Internet Explorer download option "run this program from its current location"; Entercept does not allow this option, and all our attempts were blocked by the host agent. While this rule blocked our exploit attempts, any legitimate attempts to execute non-malicious code in this manner would also be blocked. For the second test we selected the "save this program to disk" option. Entercept then prevented only 4 of our 12 web page exploits. One exploit, Iworm.mydoom.h, first deleted itself and then began emailing itself to addresses such as "support@microsoft.com". Another exploit, win32.lash.d, a virus, spawned over 900 processes, each of which then displayed a vulgar message to the screen. A third exploit, Willow.2013, actually generated an alert with the management server, however its behaviors still met our qualification as a false negative. Though the alert it generated allows the administrator some warning that the exploit occurred, Entercept was unable to prevent all of the damage caused by the exploit.

When the successful exploits were retested against the highest level of protection, Entercept still failed to block any more of them. Although it generated alerts for several, the exploits still met our criteria of a false negative. The exploits used and the alerts generated are listed below in Table 4.

	Exploit	FN1	FN2	Alerts
1	Netbus Trojan	N	N	1 Netbus Trojan Installation (Medium)
2	Trojan.Win32.virtualroot	N	N	1 System Drive Executable Modification 1 System modification in root drive
3	IISWorm.CodeGreen.a	N	N	3 IIS CodeRed idq.dll Buffer Overflow 3 IIS %u (UTF) Encoding (M)
4	Willow.2013	N	Y Y	1 IE Envelop NTVDM Execution
5	Win2k.Stream	N	Y Y	0
6	Win32.Cabanas.b	N	Y Y	1 System Executable Writing (low)
7	Win32.Ghost.1667	N	Y Y	1 System Executable Writing (low)
8	Win32.HLLO.Zori	N	N	1 System Drive Executable Modification 1 System File Modification in Root Drive
9	Win32.Lash.d	N	Y Y	0
10	Win32.Matrix.Ordy.a	N	Y Y	0
11	Win32.Redemption.b	N	Y Y	1 System Executable Writing (low)
12	Iworm.Mydoom.h	N	Y Y	0

Table 4. McAfee Entercept Web Phase

FN1 = False Negative “run this program from its current location” option

FN2 = False Negative “save this program to disk” option

4. Disk Phase

McAfee Entercept did not stop a single disk exploit; all 12 exploits succeeded. One exploit, I-worm.alanis, added a number of files to the hard drive; each file contained another copy of the virus that was executed if the file was opened. Another exploit, Worm2, simply displayed an error message to the screen every time the exploit was executed and every time the computer was restarted thereafter. Although I-worm.bagle.at's behavior generated an alert, it still met our criteria as a false negative.

When the exploits were retested against the highest level of protection Entercept did block two of them, Worm1 and Worm2. The exploits used and the alerts generated are listed in Table 5.

	Exploit	FN		Alerts
1	IIS-Worm.IIS Worm	Y	Y	1 New startup program creation (low)
2	Worm.Win32.Lovesan.a	Y	Y	1 New startup program creation (low)
3	Worm.Win32.Muma.C	Y	Y	1 New startup program creation (low)
4	Worm1	Y	N	1 CMD Tool Access (low)
5	Worm2	Y	N	1 CMD Tool Access (low)
6	Win32.Small.2280	Y	Y	0
7	Iworm.Aliz	Y	Y	0
8	Worm.win32.sasser.b	Y	Y	1 New startup program creation (low)
9	Worm.win32.welchia.g	Y	Y	0
10	I-worm.bagle.at	Y	Y	1 Entercept Agent Shielding – File Modification
11	Trojan.call911	Y	Y	0
12	I-worm.alanis	Y	Y	0

Table 5. McAfee Entercept Email Phase

FN = False Negative

5. Installation/Un-installation

Installation of both the management server and the individual host agents for Entercept was simple and straightforward. The user was required to install the management server, host agent, and a management console on the system serving as the management server. This installation was quick and relatively easy. All three programs were installed individually using a single compact disk (CD). The host agent was installed on the victim computer using the same CD.

However, we ran into several problems during the un-installation of this product. Those problems are discussed in detail in chapter 4 section E.1.

6. Manageability/Usability

All Entercept management activities were accomplished via the management console. It can only be run locally on the computer it is installed on, and is the only way to manage the individual host agents. Users are not notified of any alerts generated by their host agent. The alerts generated by the host agents are reported to the management server and can be viewed using the management console. Unfortunately, if the server is left unattended no one will know that about alerts that have been generated. When an exploit contains a known signature, Entercept can identify that exploit by name. Otherwise, Entercept alerts contain information on what malicious behavior was attempted and what files or processes the behavior was attempted on.

The management console is simple and easy to use. It is easy to access and allows all host agents to be quickly and easily managed. The intrusion-prevention system and firewall can be controlled independently from one another, and have three possible settings: disabled, warning mode, and protection mode. In warning mode, alerts are still generated and sent to the management server but no action is taken. In protection mode alerts are generated and sent to the management server and protective action is taken.

Entercept also allows the administrator to set a protection policy. These policies determine the level of protection provided by the Entercept agents. Policies can be assigned to individual agents or groups of agents. Switching between policies is easy; however, there is no clear indicator of which policy is in effect for a given system. This was not a problem for us because we had only two hosts. However, on a large heterogeneous network this could be onerous to manage. Entercept provides five pre-defined protection policies and also allows the administrator to create his own. The pre-defined policies are shown in Table 6 below. Our testing was conducted with level 2 and level 3 policies. All exploits were first tested at level 2; all false negatives were then tested again at level 3.

	High Threats	Medium Threats	Low Threats	Information
Level 1	Prevent	Ignore	Ignore	Ignore
Preparing for Level 2	Prevent	Log	Ignore	Ignore
Level 2	Prevent	Prevent	Ignore	Ignore
Preparing for Level 3	Prevent	Prevent	Log	Ignore
Level 3	Prevent	Prevent	Prevent	Log

Table 6. Entercept Protection Policies (McAfee 2005)

Host agents listed in the management server were given the same name as the computer they resided on. Over the course of our testing we had to restore the victim computer to its original state numerous times. Each time an image was restored and the agent was reloaded onto the computer, the management console reported it as a new host, even though the name was the same. To ensure there were no naming conflicts in the management server's logs, Entercept automatically numbered occurrences, as with *Inside[1]*)..

7. False Positives

A false positive was recorded repeatedly during our testing at level 2 protection every time the management console was started. It reported that "explorer.exe" was attempting to modify the management server. Two more false positives were recorded during our testing at level 3 protection: when the user attempts to access either the

Command Prompt or any of the “Administrator Tools” available through the Microsoft Windows Control Panel.

8. Throughput

The McAfee Entercept agents had no noticeable effect on their computers’ performance. However, the management console and management server do require significant computing resources. We recommend that the computer hosting the management server be used only for that purpose, and should not be responsible for hosting any other services on the network and should not also serve as someone’s desktop computer. This helps to ensure that other network resources are not impacted during an attack or other times when the management server is otherwise under a heavy load. It also allows the management server’s own protection agent to function at a high level of protection so it itself is not exploited.

9. Analysis

Overall we were not very impressed with the performance of McAfee Entercept. During our remote phase of testing, Entercept blocked all but one exploit we attempted. Unfortunately, the exploit Entercept missed created an administrator level user account on the victim computer. Once the attacker has access to an administrator level account, he or she no longer needs to use exploits, but can use legitimate services such as telnet to login to the system remotely. Additionally, this exploit, MSrpc_dcom_ms03_026, is one of the most publicized and well-known exploits. Although Entercept blocked Core Impact’s efforts to use the same exploit, it did not prevent Metasploit from successfully executing the exploit.

Entercept stopped only 8 of 36 malicious code exploits used in the disk, email, and web phases. While 10 of those 36 were viruses, which Entercept did not claim to protect against, the remaining 26 were not. We cannot explain why so many were missed. Some of the behaviors, such as mass emailing everyone in the users address book, are well-known and understood worm behaviors. Additionally, many of the successful exploits were persistent, continuing to run even after the computer was restarted. Our tests at level 3 showed that Entercept does prevent some of these efforts from becoming persistent, however.

We believe McAfee Entercept did a poor job fulfilling the claims and promises of its marketing materials, outlined in chapter 3 section C.1. Though it did prevent almost all of the remote exploits, its performance against malicious-code exploits was unsatisfactory.

McAfee Entercept did not claim to provide any protection from reconnaissance attempts, but we tested its ability anyway. This was done because Cisco Security Agent did provide claims about its ability to thwart reconnaissance efforts and we decided to also test McAfee's ability to thwart these attempts. As Table 1 shows, it was unsuccessful at detecting reconnaissance. This information obtained included things like the user names, password policies, and services currently running on the computer, including a service named "enterceptAgent". Learning exactly what protection services a victim computer is running is very valuable as it makes it easier to launch a successful attack.

C. TESTING EXPLOITS AGAINST THE CISCO SECURITY AGENT

1. Remote Phase

The Cisco Security Agent stopped all 12 remote exploits (see Table 7). However, the SQL Server CAN-2002-0649 exploit caused the SQL service to crash; the service had to be restarted by a user. A second exploit, IIS ASN.1 Big String SpNeGo, caused the operating system to crash; the user then had to reboot the computer. Although the exploits were stopped, these crashes would cause at least a temporary denial of service.

	Tool	Exploit	FN	Alerts
1	CI	IIS CGI Filename Decode	N	11 The process attempted to receive data 1 The process has triggered to many log records, messages will be suppressed for 10 minutes.
2	CI	IIS Unicode	N	1 The process attempted to receive data

	Tool	Exploit	FN	Alerts
3	CI	IIS IDA-IDQ	N	1 Self-modifying or buffer overflow code
4	CI	SQL Server Hello	N	1 Self-modifying or buffer overflow code
5	CI	MSRPC DCOM	N	1 Self-modifying or buffer overflow code 1 Current Application Attempted to execute new application
6	CI	SQL Server CAN-2002-0649	N	2 The process attempted to access 1Self-modifying or buffer overflow code
7	CI	IIS ASN.1 Big String SpNeGo	N	1 Process attempted to call exception handling routing
8	CI	MSRPC LSASS Buffer Overflow	N	1 The process attempted to communicate
9	MS	iis_nsiislog_post	N	0
10	MS	iis50_printer_overflow	N	1 The application attempted to receive data
11	MS	iis50_webdav_ntdll	N	1 Self-modifying or buffer overflow code
12	MS	Msrpc_dcom_ms03_026	N	1 Self-modifying or buffer overflow code

Table 7. Cisco Security Agent Remote Phase

CI = Core Impact MS = Metasploit FN = False Negative

2. Email Phase

The Cisco Security Agent stopped 8 of 12 email exploits (see Table 8). Before any of the exploits were executed, the Cisco Security Agent provided a warning that the “recently downloaded program may be dangerous” and asked if we still wished to run it. We answered yes and were presented with a challenge. The challenge required us to type in four upper or lowercase letters to match a visual pattern, but once we answered the challenge, we could execute the downloaded program. Since Cisco Security Agent displays this message for all untrusted traffic, regardless of whether it was malicious or benign, we deemed this to be a fair test of the product. That is, had the product only given this warning for malicious traffic, we would not have declared it a false negative. Since Cisco Security Agent does differentiate between trusted and untrusted traffic we do not believe this is a failure on the part of the product. Traffic is declared untrusted if it is not signed by a known or trusted party. The Cisco Security Agent keeps a log of all untrusted programs installed on the host system.

False negatives included Iworm.Navidad.b and Iworm.Mydoom.g whose behavior is discussed in section B.2 of this chapter. Although Iworm.Navidad.b generated three alerts, the agent did not prevent it from sending itself to the other entries in Inside's address book, even at the highest security level Cisco Security agent did not prevent any of the exploits previously identified as false negatives.

	Exploit	FN	Alerts
1	Iworm.lovegate.i	N	2 The process attempted to write
2	Iworm.Loveletter	N	10 The process attempted to write
3	Iworm.Klez.h	N	1 The process attempted to write
4	Iworm.Moodown	N	11 The process attempted to write
5	Iworm.Navidad.b	Y Y	3 The process attempted to write
6	Iworm.Netsky.d	N	2 The process attempted to write 1 Self modifying or buffer overflow code
7	Worm.win32.Chainsaw.a	Y Y	0
8	Worm.win32.Donk.c	N	2 The process attempted to write
9	Backdoor.SdBot.aa	N	4 The process attempted to write
10	Win2k.inta.1688	Y Y	0
11	Iworm.Radix.	N	2 The process attempted to write
12	Iworm.Mydoom.g	Y Y	6 The process attempted to write

Table 8. Cisco Security Agent Email Phase

FN = False Negative

3. Web Phase

We first executed the exploits using the Internet Explorer option "run this program from its current location" option. The Cisco Security Agent stopped only 2 of the 12 exploits (see Table 9). For the second test using the "save this program to disk" option, Cisco Security Agent again stopped 2 of the 12 exploits. During both tests, the Cisco Security Agent warned that the "recently download program may be dangerous" and asked if we still wanted to run it. One false negative, the Netbus Trojan, allowed us to gain control of Inside from Outside, our attacking computer. With this control we were able to send messages to Inside, control the mouse, and open and close the CD-ROM. Although the exploit was not persistent after restart, it gave the attacker control until the victim machine was restarted.

When retested at the highest security level, Cisco Security agent did prevent two of the exploits previously identified as false negatives. The alerts generated during the second test have been omitted from Table 9 as they were the same as those generated during the first; the only difference was in how many alerts were generated.

	Exploit	FN1	FN2	Alerts
1	Netbus Trojan	Y Y	Y Y	3 The process attempted to write
2	Trojan.Win32.virtualroot	Y Y	Y Y	1 The process attempted to write
3	IIS-Worm.CodeGreen.a	Y Y	Y Y	1 The process attempted to received data 1 Self Modifying or buffer overflow code
4	Willow.2013	N	Y N	0
5	Win2k.Stream	Y Y	Y Y	0
6	Win32.Cabanas.b	Y Y	Y Y	0

	Exploit	FN1	FN2	Alerts
7	Win32.Ghost.1667	Y N	Y N	11 The process attempted to write
8	Win32.HLLO.Zori	N	N	2 The process attempted to write
9	Win32.Lash.d	Y Y	Y Y	0
10	Win32.Matrix.Ordy.a	Y Y	Y Y	0
11	Win32.Redemption.b	Y Y	Y Y	11 The process attempted to write
12	Iworm.Mydoom.u	Y Y	N	1 The process attempted to write

Table 9. Cisco Security Agent Web Phase

FN1 = False Negative "run this program from its current location" option

FN2 = False Negative "save this program to disk" option.

4. Disk Phase

The Cisco Security Agent stopped 5 of our 12 disk exploits (see Table 10). In all 12 cases when we attempted to execute the exploits, the agent warned that the program executing from removable media was potentially dangerous and asked if we still wanted to run it. False negatives included Iworm.Aliz, which began emailing itself to everyone in Inside's address book, and I-worm.Alanis whose behavior is discussed in section B.4. When tested again at the highest level of protection, the Cisco Security Agent did prevent one of the exploits, Trojan.call911, from executing.

	Exploit	FN	Alerts
1	IIS-Worm.IIS Worm	N	0
2	Worm.Win32.Lovesan.a	Y Y	0
3	Worm.Win32.Muma.C	Y Y	4 The process attempted to write
4	Worm1	Y Y	0
5	Worm2	Y Y	0

	Exploit	FN	Alerts
6	Win32.Small.2280	N	0
7	Iworm.Aliz	Y Y	0
8	Worm.win32.sasser.b	N	1 The process attempted to write
9	Worm.win32.welchia.g	N	1 The process attempted to write
10	I-worm.bagle.at	N	1 The process attempted to write
11	Trojan.call911	Y N	1 The process attempted to read (low)
12	I-worm.alanis	Y Y	3 The process attempted to write

Table 10. Cisco Security Agent Disk Phase

5. Installation/Un-installation

The installation of the Cisco Security Agent was more difficult and time-consuming than the installation of McAfee Entercept. To install the management server, we also had to install some components of Cisco Works, a virtual private network program. The protection agent for the host was automatically installed along with the management server. The installation of Cisco Works and the management server took approximately one hour. For most of that hour the installation proceeded without any human intervention. Problems encountered during installation are discussed in chapter 4 section E.2.

Installation of the agents was easy. Instead of requiring the administrator to individually install the host agents using a CD, Cisco Security Agent allows the administrator to create agent kits. These kits allow the administrator to tailor which groups the new host agent will join and which predefined rule modules it will enforce. The agent kits are then created and "published", after which they can then be accessed by any computer connected to the network. To install the agent, the administrator sends the web address of the agent kit to the host users via email; the user clicks on the link, downloads, and runs the program. The installation proceeds without any other user intervention, the computer automatically reboots, and the host agent automatically registers with the management server.

Uninstallation of the Cisco Security Agent was easy. We were able to quickly uninstall the host agents, management server, and Cisco Works at the conclusion of our testing.

6. Manageability/Usability

The management center is accessed using Cisco Works as a portal. Any computer connected on the network can reach Cisco Works on the management server allowing the administrator to use any computer to receive alerts and manage the host agents.

Should the administrator choose to allow it, the user can perform some minimal management of the agent residing on his or her computer. This allows a trusted user to select the appropriate security level (high, medium, low, or off) of their host agent. However, the administrator is only notified if the user sets the security level to off, not if the level is lowered to medium or low. All users can view the alerts generated by the agent residing on their computer. We considered this a distinct advantage of Cisco's Security Agent over that of McAfee's Entercept, particularly for remote users not constantly connected to the management server. The user is notified of these alerts via a taskbar icon of a red flag that begins to "wave" when an alert is recorded. These alerts are also sent to the management server.

The warnings generated by the host agents about potentially dangerous files that have recently been downloaded could quickly become annoying. Replying to the query and answering the challenge question is not difficult, but it does take time. While the warning is a valuable reminder, especially to novice users, some users may disable or ignore security to avoid these warnings and challenges.

The management server is easy to use and to learn. The alerts generated by the host agents are easy to manage and contained detailed forensics information. Although the management server cannot identify the exploit specifically, it does provide information on what rule generated the alert, what behavior the exploit attempted, and what processes or files the exploit attempted to perform this behavior on. It also provides detailed forensic information allowing the administrator to take appropriate actions.

7. False Positives

Several false positives were encountered in our testing of Cisco Security Agent. Attempting to open the Microsoft Windows "Control Panel" via the "Windows Toolbar" resulted in one such false positive. Also, while using the Microsoft Outlook Email client and clicking on the "Send/Receive" email button on Inside resulted in a false positive on Outside. To send or receive email, we had to disable the security on the computer hosting the email server.

8. Throughput

The Cisco Security Agent's protective agent had no noticeable effect on the host computer's throughput. However, the management server required significant computing resources. We recommend the computer hosting the management server function as a dedicated management server. The reasons are the same as those outlined for McAfee Entercept in section B.8.

9. Analysis

Overall, we thought the Cisco Security Agent performed fairly well. During the reconnaissance phase of testing, we were able to determine what ports were open, the RPC endpoints, and several other pieces of information, but our efforts to access most information were thwarted. The Cisco Security Agent also prevented all 12 of our remote phase exploits. Although two of those exploits resulted in at least a temporary loss of services on the victim computer, full execution of the exploits was blocked.

The Cisco Security agent also stopped 17 of our 36 malicious code exploits. While this is far from perfect, it did stop more then twice the number of exploits that McAfee Entercept did. Additionally, only a few of the exploits were persistent after restart. Cisco Security Agent prevented the exploits from rewriting the system registries and adding new files, actions that if performed would have allowed the exploits to continue to run after a restart. However, the Cisco Security Agent provided poor protection against Trojan Horses. For example the Netbus Trojan is not a new exploit, yet it did not detect it.

Although we were not testing specifically for false positives, the two we discovered were important. Although the Cisco Security Agent allows the administrator to create, modify, and delete the rules being enforced by the host agent, this process should be avoided whenever possible. Changing the rules to prevent these false positives could have unforeseen consequences that leave the system less protected.

We believe Cisco Security Agent did an acceptable job fulfilling the claims and promises its own marketing materials made, outlined in chapter 3, section C.2. Although it did not prevent all our exploits, it did an excellent job minimizing the damage of those that were successful. Additionally, its ability to thwart reconnaissance efforts and remote exploits will greatly reduce an attacker’s chances of executing a successful remote attack.

VI. CONCLUSIONS

A. HOST-BASED INTRUSION PREVENTION SYSTEMS

Overall, host-based intrusion-prevention systems appear useful and worth the added expense. They allow administrators to optimize protection for each host on the network. They support de-parameterization (a reduced dependence on placing security measures at the perimeter of the network to provide its security) and the creation of ad-hoc, dynamic networks where perimeters are often changing or otherwise ill-defined. The “tighter” security perimeter deployed around each host prevents attacks by insiders and local users that many network-based systems would miss.

Most host-based systems perform inspections at the kernel level to prevent the attacker from using insertion or evasion attacks against the protection agents. Exploit tactics such as improper fragmentation and malformed packets will not affect a kernel-level inspection. Additionally, tactics for subverting signature-based network-based systems such as space padding will not effect an inspection based on the behavior of the application. To subvert a system, attackers must discover its behavioral rules and design an exploit that does not violate those rules. Since the protection can be tailored to each host, the attacker may have to create multiple custom exploits.

Neither of the products we evaluated prevented every exploit. But they will definitely provide better protection against insertion and evasion attacks, better protection against some exploits with no known signature, and better protection against local and insider attacks. Further improvements do not appear difficult and will surely come with time.

B. RECOMMENDATIONS

A secondary goal of this thesis was to provide the Department of Defense with a recommendation about whether to purchase these products. We base this recommendation on both the products’ performance in the tests individually and by their performance against one another. While neither system is perfect, we believe that Cisco

Security Agent's performance exceeded that of McAfee Entercept, and we recommend it alone.

McAfee Entercept performed poorly during four of our five phases of testing. In the remote phase Entercept did perform well, though it still allowed an older well-known exploit to execute successfully at medium security. Even at the highest security level, the alert generated did not properly warn the administrator of what had happened. One of Entercept's best features is its ability to identify threats using its database of known attack signatures, yet in this case the signature was not identified.

If future versions of McAfee Entercept correct the deficiencies noted in our tests and support more hosts per management server, our negative recommendation should be reconsidered.

We do recommend Cisco Security Agent for deployment by the Department of Defense. Although its performance during the web phase of testing was poor, and the performance during the disk phase was only marginal, the performance during the reconnaissance, remote, and email phases was excellent. Additionally, even when exploits were successful the damaged caused was often minimal; many times simply restarting the exploited computer eliminated any effects caused by the exploit. Also, the ability of the management server to manage up to 100,000 host agents will allow flexibility and scalability for both current and future networks.

A successful deployment of the Cisco Security Agent will require significant time, effort, and resources, but we believe such a deployment is worthwhile for important computer systems. For this to succeed, Cisco must provide assistance during the initial setup and configuration to ensure the proper rule sets are selected for enforcement. Careful planning and configuration is essential to reduce the number of false positives and false negatives.

C. PENETRATION TESTING

Decisions about purchase of information security systems should never be made without careful evaluation and testing. As this work demonstrated, simply trusting the claims made by the manufacturer is not enough. While this research tested only two host-based intrusion-prevention systems, the configuration and procedures we used could be easily replicated to test any number of products. Our research was not especially difficult or expensive. The computers we used were former lab machines that had been retired but still continued to function; our software was either an evaluation copy, open-source copy, or a licensed version. The configuration and installation of the products was accomplished with little difficulty. Without much difficulty we obtained an excellent idea of the strengths and weaknesses of both products we tested.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX: TESTING PROCEDURES

A. REMOTE EXPLOIT PROCEDURE

1. Load Core Impact
2. Create New Workspace
3. Add new host, use IP Address of victim machine
4. Click on “Attack and Penetration”
5. Click on the “Advance Tab”
6. Double click on “Exploits”
7. Double click on “Remote”
8. Click on desired exploit, using mouse drag it over to the victim’s icon and release
9. Wait for exploit to complete
10. Check management server for alerts
11. Repeat for next exploit
12. Restart as necessary to ensure services on victim machine are running.
13. After completing the Core Impact exploits open the Metasploit framework.
14. Type “use X” with X being your exploit name.
15. Type “set PAYLOAD X” with X being your payload name
16. Type “show options”
17. Fill in all required fields by typing “set VALUE X” (VALUE is the required field, X is the field’s variable)
18. Type “show targets”
19. Type “set TARGET X” where X is the target number selected from the list

20. Type “exploit”
21. Wait for exploit to complete
22. Check Management server for alerts
23. Repeat for next exploit.

B. EMAIL EXPLOIT PROCEDURE

1. Ensure your chosen email server and client are properly configured
2. Open Email Client on Outside
3. Select Create New Mail
4. Attach Malicious Code
5. Fill in subject and body as required by your testing
6. Send to victim
7. Open email client on victim
8. View Task manager to monitor processes
9. Open attachment
10. Run attachment
11. Wait one – two minutes or until processes stop, monitor for noticeable behaviors
12. Check management server for alerts
13. Repeat for next exploit.

C. DISK EXPLOIT PROCEDURE

1. Insert exploit disk into victim machine
2. Double click on “My computer”
3. Double click on “3.5” Floppy Drive”
4. Open Task M anager to monitor processes

5. Double click on exploit
6. Wait one – two minutes or until processes stop, monitor for noticeable behaviors
7. Check management server for alerts
8. Repeat for next exploit

D. WEB PHASE EXPLOIT PROCEDURE

1. Open Victim Computer's Web Browser
2. Browse to the page containing you malicious code links
3. Click on the exploit to download
4. Select the "run this program from its current location" option
5. Execute the exploit
6. Wait one – two minutes or until processes stop, monitor for noticeable behaviors
7. Check management server for alerts
8. Repeat for next exploit
9. When finished select first exploit again
10. Select the "save this program to disk" option
11. Execute the exploit
12. Wait one – two minutes or until processes stop, monitor for noticeable behaviors
13. Check management server for alerts
14. Repeat for next exploit

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

1. Robert Lemos and Dawn Kawamoto, *Sasser Variants Pose Greater Danger*, Tech Republic May 4, 2004, http://techrepublic.com.com/5100-1035_11-5205182.html# (28 May 2005)

2. Nigel Beighton *Early Alerting – The key to proactive security*, May 2004, Technews Home http://securitysa.com/article.asp?pklArticleID=2974&pklIssueID=32&pklCategoryID=11 (22 May 2005)

3. Bruce Moulton *Time for a Better Trust Infrastructure*, June 15, 2004, Symantec (Financial Services), http://enterprisesecurity.symantec.com/industry/finance/article.cfm?articleid=4114 (22 May 2005)

4. *The Internet of Today… and Tomorrow*, Symantec (Small Business), http://www.symantec.com/region/in/smallbiz/library/tomorrow.html (22 May 2005)

5. Gary Sevounts, *Distribution Utilities and Information Security*, Symantec (Power and Energy) October 1, 2004, http://enterprisesecurity.symantec.com/industry/power/article.cfm?articleid=4743&EID=0 (22 May 22, 2005)

6. Jay Beale, James Foster, and Jeffrey Posluns, *Snort 2.0 Intrusion Detection* (Rockland: Syngress 2003)

7. Paul Proctor *The Practical Intrusion Detection Handbook* (Upper Saddle River: Prentice Hall 2001) 8

8. Internet/Network Security “Stateful Inspection” About, http://netsecurity.about.com/cs/generalsecurity/g/def_stateful.htm (May 22, 2005)

9. Ellen Messmer, *Security Debate Rages*, Network World, October 2003, http://www.networkworld.com/news/2003/1006ids.html (May 22, 2005)

10. William Stallings, *Network Security Essentials: Applications and Standards*, second edition, (Upper Saddle River: Prentice Hall, 2003)

11. Shai Rubin, Somesh Jha, Barton Miller, *Automatic Generation and Analysis of NIDS Attacks*, ACSAC, 2004, http://www.cs.wisc.edu/~shai/59final.pdf (22 May 2005)

12. Thomas Ptacek, Timothy Newsham, *"Insertion Evasion and Denial of Service: Eluding Network Intrusion Detection*, January 1998, Secure Networks Inc, http://www.insecure.org/stf/secnet_ids/secnet_ids.html (22 May 2005)

13. Global Information Grid, National Security Agency : Central Security Service http://www.nsa.gov/ia/industry/gig.cfm?MenuID=10.3.2.2 (24 May 2005)

14. Entercept 5.0 Evaluation Guide

15. Cisco Security Agent Version 4.5 Data Sheet

16. Phizz0r Hackers Playground August 2001 <http://www.hackersplayground.com> (March 2005)

17. VX Heavens November 1999 <http://vx.netlux.org/> (March 2005)

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Neil Rowe
   Naval Postgraduate School
   Monterey, California

4. J.D. Fulp
   Naval Postgraduate School
   Monterey, California

5. Charles Herring
   Naval Postgraduate School
   Monterey, California

6. Greg Abelar
   Cisco Systems
   San Jose, California

7. Keith Labbe
   Naval Postgraduate School
   Monterey, California